

```
SSSSSSSSSSSSS  MMM      MMM      GGGGGGGGGGGG  RRRRRRRRRRRR  TTTTTTTTTTTTTT  LLL
SSSSSSSSSSSSS  MMM      MMM      GGGGGGGGGGGG  RRRRRRRRRRRR  TTTTTTTTTTTTTT  LLL
SSSSSSSSSSSSS  MMM      MMM      GGGGGGGGGGGG  RRRRRRRRRRRR  TTTTTTTTTTTTTT  LLL
SSS            MMMMMM  MMMMMM  GGG            RRR      RRR      TTT      LLL
SSS            MMMMMM  MMMMMM  GGG            RRR      RRR      TTT      LLL
SSS            MMMMMM  MMMMMM  GGG            RRR      RRR      TTT      LLL
SSS            MMM      MMM      GGG            RRR      RRR      TTT      LLL
SSS            MMM      MMM      GGG            RRR      RRR      TTT      LLL
SSS            MMM      MMM      GGG            RRR      RRR      TTT      LLL
SSS            MMM      MMM      GGG            RRR      RRR      TTT      LLL
SSSSSSSSSSS    MMM      MMM      GGG            RRRRRRRRRRRR  TTT      LLL
SSSSSSSSSSS    MMM      MMM      GGG            RRRRRRRRRRRR  TTT      LLL
SSSSSSSSSSS    MMM      MMM      GGG            RRRRRRRRRRRR  TTT      LLL
SSS            SSS      MMM      GGG      GGGGGGGGGG  RRR      RRR      TTT      LLL
SSS            SSS      MMM      GGG      GGGGGGGGGG  RRR      RRR      TTT      LLL
SSS            SSS      MMM      GGG      GGGGGGGGGG  RRR      RRR      TTT      LLL
SSS            SSS      MMM      GGG      GGG      GGG  RRR      RRR      TTT      LLL
SSS            SSS      MMM      GGG      GGG      GGG  RRR      RRR      TTT      LLL
SSS            SSS      MMM      GGG      GGG      GGG  RRR      RRR      TTT      LLL
SSS            SSS      MMM      GGG      GGG      GGG  RRR      RRR      TTT      LLL
SSSSSSSSSSSSS  MMM      MMM      GGGGGGGGGG  RRR      RRR      TTT      LLLLLLLLLLLLLLLL
SSSSSSSSSSSSS  MMM      MMM      GGGGGGGGGG  RRR      RRR      TTT      LLLLLLLLLLLLLLLL
SSSSSSSSSSSSS  MMM      MMM      GGGGGGGGGG  RRR      RRR      TTT      LLLLLLLLLLLLLLLL
```

```

SSSSSSSS MM MM GGGGGGG DDDDDDD I I I I I SSSSSSS DDDDDDD RRRRRRR WW WW
SSSSSSSS MM MM GGGGGGG DDDDDDD I I I I I SSSSSSS DDDDDDD RRRRRRR WW WW
SS SS MMMM MMMM GG GGGGGG DD DD DD DD DD I I SS SS DD DD RR RR WW WW
SS SS MMMM MMMM GG GGGGGG DD DD DD DD DD I I SS SS DD DD RR RR WW WW
SS SS MM MM GG GGGGGG DD DD DD DD DD I I SS SS DD DD RR RR WW WW
SSSSSS MM MM GG GGGGGG DD DD DD DD DD I I SS SS DD DD RR RR WW WW
SSSSSS MM MM GG GGGGGG DD DD DD DD DD I I SS SS DD DD RR RR WW WW
SS SS MM MM GG GGGGGG DD DD DD DD DD I I SS SS DD DD RR RR WW WW
SS SS MM MM GG GGGGGG DD DD DD DD DD I I SS SS DD DD RR RR WW WW
SSSSSS MM MM GGGGGG DDDDDDD I I I I I SSSSSSS DDDDDDD RR RR WW WW
SSSSSS MM MM GGGGGG DDDDDDD I I I I I SSSSSSS DDDDDDD RR RR WW WW

LL LL I I I I I SSSSSSS
LL LL I I I I I SSSSSSS
LL LL I I SS
LL LL I I SS
LL LL I I SS
LL LL I I SSSSSS
LL LL I I SSSSSS
LL LL I I SS
LL LL I I SS
LL LL I I SS
LLLLLLLLLL I I I I I SSSSSSS
LLLLLLLLLL I I I I I SSSSSSS

```



```
1 0001 0 MODULE SMG$DISPLAY_DRAW ( %TITLE 'Display line drawing'
2 0002 0 IDENT = '1-004' ! File: SMGDISDRW.B32 Edit: PLL1004
3 0003 0 ) =
4 0004 1 BEGIN
5 0005 1
6 0006 1 *****
7 0007 1 *
8 0008 1 * COPYRIGHT (c) 1984 BY
9 0009 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
10 0010 1 * ALL RIGHTS RESERVED.
11 0011 1 *
12 0012 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
13 0013 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
14 0014 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
15 0015 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
16 0016 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
17 0017 1 * TRANSFERRED.
18 0018 1 *
19 0019 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
20 0020 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
21 0021 1 * CORPORATION.
22 0022 1 *
23 0023 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
24 0024 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
25 0025 1 *
26 0026 1 *
27 0027 1 *****
28 0028 1
29 0029 1
30 0030 1 ++
31 0031 1 FACILITY: Screen Management
32 0032 1
33 0033 1 ABSTRACT:
34 0034 1
35 0035 1 This module contains routines that draw lines in a virtual display
36 0036 1 using the line drawing character set.
37 0037 1
38 0038 1 ENVIRONMENT: User mode - AST reentrant
39 0039 1
40 0040 1 AUTHOR: P. Levesque, CREATION DATE: 21-Sep-1983
41 0041 1
42 0042 1 MODIFIED BY:
43 0043 1
44 0044 1 1-001 - Original. PLL 21-Sep-1983
45 0045 1 1-002 - Fix to $INSERT LINE CHAR to set proper rendition when ORing
46 0046 1 line drawing characters. PLL 13-Apr-1984
47 0047 1 1-003 - Fix bug preventing drawing of one character lines. STAN 3-Jun-1984.
48 0048 1 1-004 - Don't pass SMG$C_PUT_CHARS as the function code to check for output db
49 0049 1 - this makes output believe the change is confined to 1 line. PLL 6-Sep-1984
50 0050 1 --
```

```

52 0051 1 %SBTTL 'Declarations'
53 0052 1
54 0053 1 SWITCHES:
55 0054 1
56 0055 1
57 0056 1 REQUIRE 'RTLIN:SMGPROLOG';
58 0134 1
59 0135 1 REQUIRE 'RTLIN:STRLNK';          ! JSB linkage for string routines
60 0320 1
61 0321 1 LINKAGES:
62 0322 1
63 0323 1 NONE
64 0324 1
65 0325 1 TABLE OF CONTENTS:
66 0326 1
67 0327 1
68 0328 1 FORWARD ROUTINE
69 0329 1 SMG$DRAW_LINE,
70 0330 1 SMG$DRAW_RECTANGLE;
71 0331 1
72 0332 1
73 0333 1 INCLUDE FILES:
74 0334 1
75 0335 1
76 0336 1
77 0337 1
78 0338 1 MACROS:
79 0339 1
80 0340 1 +
81 0341 1 The following macro inserts a line drawing character into the text
82 0342 1 buffer and checks to see if the previous character was a line drawing
83 0343 1 character. If it was, it ORs in the new character so that intersecting
84 0344 1 lines will appear correctly on the screen. Notice that although border
85 0345 1 elements are line drawing characters, we ignore those here - we only want
86 0346 1 proper intersections with other lines the user drew.
87 0347 1
88 0348 1 MACRO
89 M 0349 1 $INSERT_LINE_CHAR (CHAR) =
90 M 0350 1 BEGIN
91 M 0351 1 IF .(ATTR_BUF [.POS])<ATTR_V_USER_GRAPHIC, 1>
92 M 0352 1 THEN
93 M 0353 1 BEGIN
94 M 0354 1 TEXT_BUF [.POS] = .TEXT_BUF [.POS] OR CHAR;
95 M 0355 1 ATTR_BUF [.POS] = .ATTR_BUF [.POS] AND
96 M 0356 1 (NOT (SMG$M_BLINK+SMG$M_BOLD+SMG$M_REVERSE+SMG$M_UNDERLINE))
97 M 0357 1 OR .REND_CODE;
98 M 0358 1 END
99 M 0359 1 ELSE
100 M 0360 1 BEGIN
101 M 0361 1 TEXT_BUF [.POS] = CHAR;
102 M 0362 1 ATTR_BUF [.POS] = ATTR_M_USER_GRAPHIC OR .REND_CODE;
103 M 0363 1 END;
104 M 0364 1 END%;
105 0365 1
106 0366 1 !+
107 0367 1 ! This macro resets a line to single wide/high. The line must be
108 0368 1 ! blanked first.

```



```

: 109      0369 1  !-
: 110      0370 1  MACRO
: 111      M 0371 1  $RESET_LINE (LINE_NO) =
: 112      M 0372 1  BEGIN ! reset/blank line
: 113      M 0373 1  LOCAL
: 114      M 0374 1  START_INDEX;
: 115      M 0375 1  START_INDEX = $SMG$LINEAR (LINE_NO, .DCB [DCB_W COL START]);
: 116      M 0376 1  $SMG$BLANK_FILL_DCB (.DCB [DCB_W_NO_COLS], .START_INDEX);
: 117      M 0377 1  LINE_CHAR [LINE_NO] = 0;
: 118      0378 1  END%; ! reset/blank line
: 119      0379 1
: 120      0380 1
: 121      0381 1  EQUATED SYMBOLS:
: 122      0382 1
: 123      0383 1  NONE
: 124      0384 1
: 125      0385 1  FIELDS:
: 126      0386 1
: 127      0387 1  NONE
: 128      0388 1
: 129      0389 1  PSECTS:
: 130      0390 1
: 131      0391 1
: 132      0392 1
: 133      0393 1  OWN STORAGE:
: 134      0394 1
: 135      0395 1  NONE
: 136      0396 1
: 137      0397 1  EXTERNAL REFERENCES:
: 138      0398 1
: 139      0399 1
: 140      0400 1  EXTERNAL ROUTINE
: 141      0401 1  SMG$$CHECK_FOR_OUTPUT_DCB; ! check if time to repaint display
: 142      0402 1
: 143      0403 1  EXTERNAL LITERAL
: 144      0404 1  SMG$_INVDIS_ID, ! Invalid display id
: 145      0405 1  SMG$_INVARG, ! Invalid argument
: 146      0406 1  SMG$_INVCOL, ! Invalid column number
: 147      0407 1  SMG$_INVROW, ! Invalid row number
: 148      0408 1  SMG$_DIALINNOT, ! Diagonal line not allowed
: 149      0409 1  SMG$_WRONUMARG; ! Wrong number of arguments

```

```

151 0410 1 %SBTTL 'SMG$DRAW_LINE - Draw a line in a virtual display'
152 0411 1 GLOBAL ROUTINE SMG$DRAW_LINE (
153 0412 1
154 0413 1 DISPLAY ID,
155 0414 1 START_ROW,
156 0415 1 START_COL,
157 0416 1 END_ROW,
158 0417 1 END_COL,
159 0418 1 RENDITION_SET,
160 0419 1 RENDITION_COMPLEMENT
161 0420 1 ) =
162 0421 1 ++
163 0422 1 FUNCTIONAL DESCRIPTION:
164 0423 1
165 0424 1 This routine draws a line from START_ROW, START_COL to END_ROW, END_COL.
166 0425 1 Device independent codes are moved to the display text and
167 0426 1 attributes buffers - output later figures out what characters
168 0427 1 to use based on the type of terminal.
169 0428 1
170 0429 1 The internal cursor position is not changed.
171 0430 1
172 0431 1 CALLING SEQUENCE:
173 0432 1
174 0433 1 ret_status.wlc.v = SMG$DRAW_LINE (DISPLAY_ID.rl.r,
175 0434 1 START_ROW.rl.r, START_COL.rl.r,
176 0435 1 END_ROW.rl.r, END_COL.rl.r
177 0436 1 [,RENDITION_SET.rl.r]
178 0437 1 [,RENDITION_COMPLEMENT.rl.r])
179 0438 1
180 0439 1 FORMAL PARAMETERS:
181 0440 1
182 0441 1 DISPLAY_ID.rl.r Display id of virtual display
183 0442 1
184 0443 1 START_ROW.rl.r Address of row number at which
185 0444 1 to start line.
186 0445 1
187 0446 1 START_COL.rl.r Address of column number at which
188 0447 1 to start line.
189 0448 1
190 0449 1 END_ROW.rl.r Address of row number at which
191 0450 1 to end line.
192 0451 1
193 0452 1 END_COL.rl.r Address of column number at which
194 0453 1 to end line.
195 0454 1
196 0455 1 RENDITION_SET.rl.r Optional. Each 1 bit in this parameter
197 0456 1 causes the corresponding attribute to be
198 0457 1 set in the display. (See below for list
199 0458 1 of settable attributes.)
200 0459 1
201 0460 1 RENDITION_COMPLEMENT.rl.r Optional. Each 1 bit attribute in this
202 0461 1 parameter causes the corresponding attribute
203 0462 1 to be complemented in the display. (See
204 0463 1 below for list of complementable attributes.)
205 0464 1
206 0465 1 If the same bit is specified in both the RENDITION_SET parameter
207 0466 1 and in the RENDITION_COMPLEMENT parameter, the application is

```



```

208 0467 1 1  RENDITION_SET followed by RENDITION complement. Using these two
209 0468 1 1  parameters together the caller can exercise arbitrary and
210 0469 1 1  independent control over each attribute on a single call. On an
211 0470 1 1  attribute by attribute basis he can cause the following
212 0471 1 1  transformations:
213 0472 1 1
214 0473 1 1
215 0474 1 1
216 0475 1 1
217 0476 1 1
218 0477 1 1
219 0478 1 1
220 0479 1 1
221 0480 1 1
222 0481 1 1
223 0482 1 1
224 0483 1 1
225 0484 1 1
226 0485 1 1
227 0486 1 1
228 0487 1 1
229 0488 1 1
230 0489 1 1
231 0490 1 1
232 0491 1 1
233 0492 1 1
234 0493 1 1
235 0494 1 1
236 0495 1 1
237 0496 1 1
238 0497 1 1
239 0498 1 1
240 0499 1 1
241 0500 1 1
242 0501 1 1
243 0502 1 1
244 0503 1 1
245 0504 1 1
246 0505 1 1
247 0506 1 1
248 0507 1 1
249 0508 1 1
250 0509 1 1
251 0510 1 1
252 0511 1 1
253 0512 1 1
254 0513 1 1
255 0514 1 1
256 0515 2 2
257 0516 2 2
258 0517 2 2
259 0518 2 2
260 0519 2 2
261 0520 2 2
262 0521 2 2
263 0522 2 2
264 0523 2 2

```

Attributes which can be manipulated in this manner are:

SMG\$M_BLINK displays characters blinking.

SMG\$M_BOLD displays characters in higher-than-normal intensity.

SMG\$M_REVERSE displays characters in reverse video -- that is, using the opposite default rendition of the virtual display.

SMG\$M_UNDERLINE displays characters underlined.

IMPLICIT INPUTS:

NONE

IMPLICIT OUTPUTS:

NONE

COMPLETION STATUS:

SS\$ NORMAL Normal successful completion

SMG\$_INVCOL Invalid column number

SMG\$_INVROW Invalid row number

SMG\$_DIALINNOT Diagonal line not allowed

SMG\$_WRONUMARG Wrong number of arguments

SIDE EFFECTS:

NONE

--

BEGIN

LOCAL

WORK_S_ROW, : local temporaries

WORK_S_COL, : ..

WORK_E_ROW, : ..

WORK_E_COL, : ..

TEXT_BUF : REF VECTOR [,BYTE], : ptr to dcb text buffer

ATTR_BUF : REF VECTOR [,BYTE], : ptr to dcb attr buffer


```

265      0524 2      DIR : INITIAL (0),      ! direction of line
266      0525 2      REND_CODE,      ! rendition code to use
267      0526 2      DCB : REF BLOCK [,BYTE];      ! addr of display control block
268      0527 2
269      0528 2      LITERAL
270      0529 2      K_SET_ARG = 6,      ! arg number of rend set
271      0530 2      K_COMP_ARG = 7,      ! arg number of rend complement
272      0531 2      K_HORIZ = 1;      ! line is horizontal
273      0532 2
274      0533 2
275      0534 2      $SMG$GET_DCB (.DISPLAY_ID, DCB);      ! get addr of virtual display
276      0535 2      ! control block
277      0536 2      TEXT_BUF = .DCB [DCB_A_TEXT_BUF];
278      0537 2      ATTR_BUF = .DCB [DCB_A_ATTR_BUF];
279      0538 2
280      0539 2      +
281      0540 2      - Validate arguments passed. Check for optionals.
282      0541 2
283      0542 2
284      0543 2      $SMG$VALIDATE_ARGCOUNT (5, 7);
285      0544 2
286      0545 2      $SMG$VALIDATE_ROW_COL (..START_ROW, ..START_COL);
287      0546 2
288      0547 2      $SMG$VALIDATE_ROW_COL (..END_ROW, ..END_COL);
289      0548 2
290      0549 2      $SMG$SET_REND_CODE (K_SET_ARG, K_COMP_ARG);
291      0550 2      ! macro to use caller's args if present
292      0551 2
293      0552 2      +
294      0553 2      - Use local temporary storage for start and end points. We don't know
295      0554 2      the order in which the points will be passed, but our code assumes
296      0555 2      left to right order for horizontal lines, and up to down order for vertical
297      0556 2      lines.
298      0557 2
299      0558 2
300      0559 2      IF ..START_ROW EQL ..END_ROW
301      0560 2      THEN
302      0561 2      DIR = K_HORIZ
303      0562 2      ELSE
304      0563 2      IF ..START_COL NEQ ..END_COL
305      0564 2      THEN
306      0565 2      RETURN (SMG$_DIALINNOT);
307      0566 2
308      0567 2      IF .DIR EQL K_HORIZ
309      0568 2      THEN
310      0569 2      BEGIN
311      0570 2      IF ..START_COL LSS ..END_COL
312      0571 2      THEN
313      0572 2      BEGIN
314      0573 2      WORK_S_COL = ..START_COL;
315      0574 2      WORK_S_ROW = ..START_ROW;
316      0575 2      WORK_E_COL = ..END_COL;
317      0576 2      WORK_E_ROW = ..END_ROW;
318      0577 2      END
319      0578 2      ELSE
320      0579 2      BEGIN
321      0580 2      WORK_S_COL = ..END_COL;

```



```

322      0581  4      WORK_S_ROW = ..END_ROW;
323      0582  4      WORK_E_COL = ..START_COL;
324      0583  4      WORK_E_ROW = ..START_ROW;
325      0584  3      END;
326      0585  3      END
327      0586  2      ELSE ! vertical line
328      0587  2      IF ..START_ROW LSS ..END_ROW
329      0588  2      THEN
330      0589  3      BEGIN
331      0590  3      WORK_S_COL = ..START_COL;
332      0591  3      WORK_S_ROW = ..START_ROW;
333      0592  3      WORK_E_COL = ..END_COL;
334      0593  3      WORK_E_ROW = ..END_ROW;
335      0594  3      END
336      0595  2      ELSE
337      0596  3      BEGIN
338      0597  3      WORK_S_COL = ..END_COL;
339      0598  3      WORK_S_ROW = ..END_ROW;
340      0599  3      WORK_E_COL = ..START_COL;
341      0600  3      WORK_E_ROW = ..START_ROW;
342      0601  2      END;
343      0602  2
344      0603  2      !+
345      0604  2      Reset the line characteristics vector in case there was previously
346      0605  2      some double high or double wide text.
347      0606  2
348      0607  2      We may be dealing with a vertical or horizontal line. A vertical line
349      0608  2      will require re-setting multiple entries in the vector.
350      0609  2      -
351      0610  2
352      0611  3      BEGIN
353      0612  3      BIND
354      0613  3      LINE_CHAR = .DCB [DCB_A_LINE_CHAR];
355      0614  3      MAP
356      0615  3      LINE_CHAR : VECTOR [,BYTE];
357      0616  3
358      0617  3      IF .DIR EQL K_HORIZ ! horiz line - reset 1 ele of vector
359      0618  3      THEN
360      0619  4      BEGIN
361      0620  4      IF .LINE_CHAR [.WORK_S_ROW] NEQ 0
362      0621  4      THEN
363      0622  4      $RESET_LINE (.WORK_S_ROW);
364      0623  4      END
365      0624  3      ELSE
366      0625  4      BEGIN ! vert line - reset multi elements
367      0626  4      INCR ROW FROM .WORK_S_ROW TO .WORK_E_ROW DO
368      0627  5      BEGIN
369      0628  5      IF .LINE_CHAR [.ROW] NEQ 0
370      0629  5      THEN
371      0630  5      $RESET_LINE (.ROW);
372      0631  4      END;
373      0632  3      END;
374      0633  3
375      0634  2      END;
376      0635  2
377      0636  2      !+
378      0637  2      Insert the generic text byte for line drawing into the text buffer.

```



```
: 379      0638 2 | (Notice that there are different representations for horizontal, vertical,
: 380      0639 2 | corners, etc.) Set a bit in the attribute byte so the output routines
: 381      0640 2 | will re-interpret the generic text into the device specific character.
: 382      0641 2 |
: 383      0642 2 | We move one character at a time since it may be necessary to OR elements
: 384      0643 2 | (see description of macro above).
: 385      0644 2 |
: 386      0645 2 |
: 387      0646 2 | BEGIN
: 388      0647 2 | LOCAL
: 389      0648 2 |     POS,                                ! linear index into text buffer
: 390      0649 2 |     LENGTH;                            ! calc length of line
: 391      0650 2 |
: 392      0651 2 | POS = $SMG$LINEAR (.WORK_S_ROW, .WORK_S_COL);
: 393      0652 2 |
: 394      0653 2 | IF .DIR EQL K_HORIZ                    ! horizontal line
: 395      0654 2 | THEN
: 396      0655 4 |     BEGIN
: 397      0656 4 |     LENGTH = .WORK_E_COL - .WORK_S_COL + 1;
: 398      0657 4 |
: 399      0658 4 |     $INSERT_LINE_CHAR (BORD_M_RIGHT);
: 400      0659 4 |     POS = .POS + 1;
: 401      0660 4 |
: 402      0661 4 |     INCR NUM_CHAR FROM 2 TO (.LENGTH - 1) DO
: 403      0662 5 |     BEGIN
: 404      0663 5 |     $INSERT_LINE_CHAR (BORD_M_HORIZ);
: 405      0664 5 |     POS = .POS + 1;
: 406      0665 5 |     END;
: 407      0666 4 |
: 408      0667 4 |     IF .LENGTH GTR 1
: 409      0668 4 |     THEN
: 410      0669 4 |     $INSERT_LINE_CHAR (BORD_M_LEFT);
: 411      0670 4 |
: 412      0671 4 |     END
: 413      0672 3 | ELSE
: 414      0673 4 |     BEGIN                                ! vertical line
: 415      0674 4 |     LENGTH = .WORK_E_ROW - .WORK_S_ROW + 1;
: 416      0675 4 |
: 417      0676 4 |     $INSERT_LINE_CHAR (BORD_M_DOWN);
: 418      0677 4 |     POS = .POS + .DCB [DCB_W_NO_COLS];
: 419      0678 4 |
: 420      0679 4 |     INCR ROW FROM 2 TO (.LENGTH - 1) DO
: 421      0680 5 |     BEGIN
: 422      0681 5 |     $INSERT_LINE_CHAR (BORD_M_VERT);
: 423      0682 5 |     POS = .POS + .DCB [DCB_W_NO_COLS];
: 424      0683 5 |     END;
: 425      0684 4 |
: 426      0685 4 |     IF .LENGTH GTR 1
: 427      0686 4 |     THEN
: 428      0687 4 |     $INSERT_LINE_CHAR (BORD_M_UP);
: 429      0688 4 |
: 430      0689 3 |     END;
: 431      0690 3 |
: 432      0691 2 | END;
: 433      0692 2 |
: 434      0693 2 | !+
: 435      0694 2 | ! See if this change should be reflected on the screen immediately.
```


SMG\$DISPLAY_DRA Display line drawing
1-004

SMG\$DRAW_LINE - Draw a line in a virtual displa

F 8
16-Sep-1984 00:24:52
14-Sep-1984 13:09:40

VAX-11 Bliss-32 V4.0-742
[SMGRTL.SRC]SMGDISDRW.B32;2

Page 9
(3)

```
: 436      0695  2  !-  
: 437      0696  2  
: 438      0697  2  
: 439      0698  2  
: 440      0699  2  
: 441      0700  2  
: 442      0701  2  
: 443      0702  1  
      RETURN (SMG$$CHECK_FOR_OUTPUT_DCB (.DCB,  
      0,  
      .WORK_S_ROW));  
      END;  
      ! End of routine SMG$DRAW_LINE
```

.TITLE SMG\$DISPLAY_DRAW Display line drawing
.IDENT \1-004\

.EXTRN SMG\$\$CHECK_FOR_OUTPUT_DCB
.EXTRN SMG\$_INVDIS_ID, SMG\$_INVARG
.EXTRN SMG\$_INVCOL, SMG\$_INVROW
.EXTRN SMG\$_DIALINNOT, SMG\$_WRONUMARG

.PSECT _SMG\$CODE, NOWRT, SHR, PIC, 2

.ENTRY SMG\$DRAW_LINE, Save R2,R3,R4,R5,R6,R7,R8,-
R9,R10,RT1

SUBL2 #32, SP
CLRL DIR
MOVL @DISPLAY_ID, R0
CML 56(R0), @DISPLAY_ID
BNEQ 1\$
CMPB 68(R0), #17
BEQL 2\$
MOVL #SMG\$_INVDIS_ID, R0
RET
MOVL @DISPLAY_ID, DCB
MOVL 16(DCB), TEXT_BUF
MOVL 20(DCB), ATTR_BUF
SUBB3 #5, (AP), DIFF
CMPB DIFF, #2
BLEQU 3\$
MOVL #SMG\$_WRONUMARG, R0
RET
MOVL @START_ROW, R3
BLEQ 4\$
CMPZV #0, #16, 2(DCB), R3
BGEQ 5\$
MOVL #SMG\$_INVROW, R0
RET
MOVL @START_COL, R2
BLEQ 6\$
CMPZV #0, #16, 6(DCB), R2
BGEQ 7\$
MOVL #SMG\$_INVCOL, R0
RET
MOVL @END_ROW, R1
BLEQ 8\$
CMPZV #0, #16, 2(DCB), R1
BGEQ 9\$
MOVL #SMG\$_INVROW, R0
RET

0411
0515
0534
0536
0537
0543
0545
0547

OFFC 00000
5E 20 C2 00002
54 D4 00005
04 50 04 BC D0 00007
BC 38 A0 D1 0000B
11 44 A0 91 00012
08 13 00016
50 00000000G 8F D0 00018 1\$:
04 5B 04 BC D0 00020 2\$:
AE 10 AB D0 00024
5A 14 AB D0 00029
6C 05 83 0002D
02 50 91 00031
08 1B 00034
50 00000000G 8F D0 00036
04 0003D
53 08 BC D0 0003E 3\$:
10 00 15 00042
08 ED 00044
08 18 0004A
50 00000000G 8F D0 0004C 4\$:
04 00053
52 0C BC D0 00054 5\$:
10 08 15 00058
00 ED 0005A
08 18 00060
50 00000000G 8F D0 00062 6\$:
04 00069
51 10 BC D0 0006A 7\$:
10 08 15 0006E
00 ED 00070
08 18 00076
50 00000000G 8F D0 00078 8\$:
04 0007F

53 02 AB
52 06 AB
51 02 AB

			50	14	BC	D0	00080	9\$:	MOVL	@END_COL, R0		
					08	15	00084		BLEQ	10\$		
50	06	AB	10		00	ED	00086		CMPZV	#0, #16, 6(DCB), R0		
					08	18	0008C		BGEQ	11\$		
			50	00000000G	8F	D0	0008E	10\$:	MOVL	#SMG\$_INVCOL, R0		
					04	00	00095		RET			
10	AE		2E	AB	9A	00096	11\$:	MOVZBL	46(DCB), REND_CODE		0549	
	06			6C	91	0009B		CMPB	(AP), #6			
				0A	1F	0009E		BLSSU	12\$			
				18	AC	D5	000A0		TSTL	24(AP)		
				05	13	000A3		BEQL	12\$			
10	AE		18	BC	C8	000A5		BISL2	@RENDITION_SET, REND_CODE			
	07			6C	91	000AA	12\$:	CMPB	(AP), #7			
				0A	1F	000AD		BLSSU	13\$			
				1C	AC	D5	000AF		TSTL	28(AP)		
				05	13	000B2		BEQL	13\$			
10	AE		1C	BC	CC	000B4		XORL2	@RENDITION_COMPLEMENT, REND_CODE			
	51			53	D1	000B9	13\$:	CMPL	R3, R1		0559	
				05	12	000BC		BNEQ	14\$			
			54	01	D0	000BE		MOVL	#1, DIR		0561	
				0D	11	000C1		BRB	15\$			
			50	52	D1	000C3	14\$:	CMPL	R2, R0		0563	
				08	13	000C6		BEQL	15\$			
			50	00000000G	8F	D0	000C8		MOVL	#SMG\$_DIALINNOT, R0		0565
					04	00	000CF		RET			
				1C	AE	D4	000D0	15\$:	CLRL	28(SP)		0567
	01			54	D1	000D3		CMPL	DIR, #1			
				08	12	000D6		BNEQ	16\$			
				1C	AE	D6	000D8		INCL	28(SP)		
			50	52	D1	000DB		CMPL	R2, R0		0570	
				03	11	000DE		BRB	17\$			
			51	53	D1	000E0	16\$:	CMPL	R3, R1		0587	
				0F	18	000E3	17\$:	BGEQ	18\$			
14	AE			52	D0	000E5		MOVL	R2, WORK_S_COL		0590	
	57			53	D0	000E9		MOVL	R3, WORK_S_ROW		0591	
	59			50	D0	000EC		MOVL	R0, WORK_E_COL		0592	
	58			51	D0	000EF		MOVL	R1, WORK_E_ROW		0593	
				0D	11	000F2		BRB	19\$		0587	
14	AE			50	D0	000F4	18\$:	MOVL	R0, WORK_S_COL		0597	
	57			51	D0	000F8		MOVL	R1, WORK_S_ROW		0598	
	59			52	D0	000FB		MOVL	R2, WORK_E_COL		0599	
	58			53	D0	000FE		MOVL	R3, WORK_E_ROW		0600	
	54							BLBC	28(SP), 22\$		0620	
				1C	AE	E9	00101	19\$:	TSTB	@76(DCB)[WORK_S_ROW]		
				4C	BB47	95	00105		BEQL	21\$		
					4C	13	00109		MOVAB	-1(R7), R6		0622
	56			FF	A7	9E	0010B		MOVZWL	6(DCB), R0		
	50			06	AB	3C	0010F		MULL2	R0, R6		
	56				50	C4	00113		MOVZWL	4(DCB), R0		
	50			04	AB	3C	00116		MOVAB	-1(R0)[R6], START_INDEX		
18	AE			FF	A046	9E	0011A		MOVL	16(DCB), TEXT_BUF		
	50			10	AB	D0	00120		MOVL	20(DCB), ATTR_BUF		
	56			14	AB	D0	00124		MOVL	24(DCB), CHAR_BUF		
0C	AE			18	AB	D0	00128		MOVC5	#0, (SP), #32, 6(DCB), @START_INDEX-		
06	AB		20		00	2C	0012D			[TEXT_BUF]		
	6E			18	BE40		00133		MOVC5	#0, (SP), 46(DCB), 6(DCB), @START_INDEX-		
06	AB		2E	AB	6E	00	00136			[ATTR_BUF]		
					18	BE46	0013D					

06	AB	30	AB	56	0C	AE	D5	00140	TSTL	CHAR_BUF	:	
				6E		0E	13	00143	BEQL	20\$:	
					0C	AE	D0	00145	MOVL	CHAR_BUF, R6	:	
						00	2C	00149	MOVCS	#0, (SP), 48(DCB), 6(DCB), @START_INDEX[R6]	:	
					18	BE46		00150			:	
					4C	BB47	94	00153	20\$:	CLRB	@76(DCB)[WORK_S_ROW]	
						5F	11	00157	21\$:	BRB	26\$	0617
				56		A7	9E	00159	22\$:	MOVAB	-1(R7), ROW	0626
						55	11	0015D		BRB	25\$	
					4C	BB46	95	0015F	23\$:	TSTB	@76(DCB)[ROW]	0628
						4F	13	00163		BEQL	25\$	
				50		A6	9E	00165		MOVAB	-1(R6), R0	0630
				51	06	AB	3C	00169		MOVZWL	6(DCB), R1	
				50		51	C4	0016D		MULL2	R1, R0	
				51	04	AB	3C	00170		MOVZWL	4(DCB), R1	
				18	FF	A140	9E	00174		MOVAB	-1(R1)[R0], START_INDEX	
				50	10	AB	D0	0017A		MOVL	16(DCB), TEXT_BUF	
				08	14	AB	7D	0017E		MOVQ	20(DCB), ATTR_BUF	
				6E		00	2C	00183		MOVCS	#0, (SP), #32, 6(DCB), @START_INDEX-	
					18	BE40		00189			[TEXT_BUF]	
					08	AE	C1	0018C		ADDL3	ATTR_BUF, START_INDEX, (SP)	
						00	2C	00192		MOVCS	#0, (SP), 46(DCB), 6(DCB), @0(SP)	
					00	BE		00199				
					0C	AE	D5	0019B		TSTL	CHAR_BUF	
						10	13	0019E		BEQL	24\$	
					0C	AE	C1	001A0		ADDL3	CHAR_BUF, START_INDEX, 8(SP)	
						00	2C	001A7		MOVCS	#0, (SP), 48(DCB), 6(DCB), @8(SP)	
					08	BE		001AE				
					4C	BB46	94	001B0	24\$:	CLRB	@76(DCB)[ROW]	
						58	F3	001B4	25\$:	AOBLEQ	WORK_E_ROW, ROW, 23\$	0626
						A7	9E	001B8	26\$:	MOVAB	-1(R7), R6	0651
					06	AB	3C	001BC		MOVZWL	6(DCB), R0	
						50	C4	001C0		MULL2	R0, R6	
						01	C3	001C3		SUBL3	#1, WORK_S_COL, R1	
						51	C1	001C8		ADDL3	R1, R6, POS	
						50	C1	001CC		ADDL3	POS, ATTR_BUF, R2	0658
					04	AE	C1	001D0		ADDL3	TEXT_BUF, POS, R3	
					1C	AE	E8	001D5		BLBS	28(SP), 27\$	
					008A	31	001D9		BRW	36\$		
					14	AE	C2	001DC	27\$:	SUBL2	WORK_S_COL, R9	0656
					01	A9	9E	001E0		MOVAB	1(R9), LENGTH	
						06	E1	001E4		BBC	#6, (R2), 28\$	0658
						01	88	001E8		BISB2	#1, (R3)	
						62	9A	001EB		MOVZBL	(R2), R3	
						0F	CA	001EE		BICL2	#15, R3	
					10	AE	89	001F1		BISB3	REND_CODE, R3, (R2)	
						09	11	001F6		BRB	29\$	
						01	90	001F8	28\$:	MOVB	#1, (R3)	
					40	8F	89	001FB		BISB3	#64, REND_CODE, (R2)	
						50	D6	00201	29\$:	INCL	POS	0659
						01	D0	00203		MOVL	#1, NUM_CHAR	0661
						2D	11	00206		BRB	33\$	
						06	E1	00208	30\$:	BBC	#6, (POS)[ATTR_BUF], 31\$	0663
					04	AE	D0	0020D		MOVL	TEXT_BUF, R2	
						05	88	00211		BISB2	#5, (POS)[R2]	
					604A	9A	00215		MOVZBL	(POS)[ATTR_BUF], R2		
						0F	CA	00219		BICL2	#15, R2	

604A	52	10	AE	89	0021C	BISB3	REND_CODE, R2, (POS)[ATTR_BUF]	:	
			OF	11	00222	BRB	32\$:	
	52		04	AE	D0	00224	31\$: MOVL	TEXT_BUF, R2	
	6042		05	90	00228	MOVB	#5, (POS)[R2]	:	
604A	10	AE	40	8F	89	0022C	BISB3	#64, REND_CODE, (POS)[ATTR_BUF]	
			50	D6	00233	32\$: INCL	POS	:	0664
CF	53		51	F2	00235	33\$: AOBLS	LENGTH, NUM_CHAR, 30\$:	0661
	01		51	D1	00239	CMPL	LENGTH, #1	:	0667
			1A	15	0023C	BLEQ	34\$:	
18	604A		06	E1	0023E	BBC	#6, (POS)[ATTR_BUF], 35\$:	0669
	51		04	AE	D0	00243	MOVL	TEXT_BUF, R1	
	6041		04	88	00247	BISB2	#4, (POS)[R1]	:	
	52		604A	9A	0024B	MOVZBL	(POS)[ATTR_BUF], R2	:	
	52		OF	CA	0024F	BICL2	#15, R2	:	
604A	52	10	AE	89	00252	BISB3	REND_CODE, R2, (POS)[ATTR_BUF]	:	
			00A1	31	00258	34\$: BRW	45\$:	
	51		04	AE	D0	0025B	35\$: MOVL	TEXT_BUF, R1	
	6041		04	90	0025F	MOVB	#4, (POS)[R1]	:	
			008F	31	00263	BRW	44\$:	
	58		57	C2	00266	36\$: SUBL2	WORK_S_ROW, R8	:	0674
	51		01	A8	9E	00269	MOVAB	1(R8), LENGTH	
10	62		06	E1	0026D	BBC	#6, (R2), 37\$:	0676
	63		08	88	00271	BISB2	#8, (R3)	:	
	53		62	9A	00274	MOVZBL	(R2), R3	:	
	53		OF	CA	00277	BICL2	#15, R3	:	
62	53	10	AE	89	0027A	BISB3	REND_CODE, R3, (R2)	:	
			09	11	0027F	BRB	38\$:	
	63		08	90	00281	37\$: MOVB	#8, (R3)	:	
62	10	AE	40	8F	89	00284	BISB3	#64, REND_CODE, (R2)	
	52		06	AB	3C	0028A	38\$: MOVZWL	6(DCB), R2	0677
	50		52	C0	0028E	ADDL2	R2, POS	:	
	53		01	D0	00291	MOVL	#1, ROW	:	0679
			32	11	00294	BRB	42\$:	
17	604A		06	E1	00296	39\$: BBC	#6, (POS)[ATTR_BUF], 40\$:	0681
	52		04	AE	D0	0029B	MOVL	TEXT_BUF, R2	
	6042		0A	88	0029F	BISB2	#10, (POS)[R2]	:	
	52		604A	9A	002A3	MOVZBL	(POS)[ATTR_BUF], R2	:	
	52		OF	CA	002A7	BICL2	#15, R2	:	
604A	52	10	AE	89	002AA	BISB3	REND_CODE, R2, (POS)[ATTR_BUF]	:	
			OF	11	002B0	BRB	41\$:	
	52		04	AE	D0	002B2	40\$: MOVL	TEXT_BUF, R2	
	6042		0A	90	002B6	MOVB	#10, (POS)[R2]	:	
604A	10	AE	40	8F	89	002BA	BISB3	#64, REND_CODE, (POS)[ATTR_BUF]	
	52		06	AB	3C	002C1	41\$: MOVZWL	6(DCB), R2	0682
	50		52	C0	002C5	ADDL2	R2, POS	:	
CA	53		51	F2	002C8	42\$: AOBLS	LENGTH, ROW, 39\$:	0679
	01		51	D1	002CC	CMPL	LENGTH, #1	:	0685
			2B	15	002CF	BLEQ	45\$:	
17	604A		06	E1	002D1	BBC	#6, (POS)[ATTR_BUF], 43\$:	0687
	51		04	AE	D0	002D6	MOVL	TEXT_BUF, R1	
	6041		02	88	002DA	BISB2	#2, (POS)[R1]	:	
	51		604A	9A	002DE	MOVZBL	(POS)[ATTR_BUF], R1	:	
	51		OF	CA	002E2	BICL2	#15, R1	:	
604A	51	10	AE	89	002E5	BISB3	REND_CODE, R1, (POS)[ATTR_BUF]	:	
			OF	11	002EB	BRB	45\$:	
	51		04	AE	D0	002ED	43\$: MOVL	TEXT_BUF, R1	
	6041		02	90	002F1	MOVB	#2, (POS)[R1]	:	

SMG\$DISPLAY_DRA	Display line drawing	J 8	16-Sep-1984 00:24:52	VAX-11 Bliss-32 V4.0-742	
1-004	SMG\$DRAW_LINE - Draw a line in a virtual displa	14-Sep-1984 13:09:40	[SMGRTL.SRC]SMGDISDRW.B32;2		
604A	10 AE	40	8F 89 002F5 44\$:	BISB3	#64, REND CODE, (POS)[ATTR_BUF]
			57 DD 002FC 45\$:	PUSHL	WORK_S_ROW
			7E D4 002FE	CLRL	-(SPT)
			5B DD 00300	PUSHL	DCB
00000000G	00		03 FB 00302	CALLS	#3, SMG\$\$CHECK_FOR_OUTPUT_DCB
			04 00309	RET	

Page 13
(3)

: 0699
: 0697
:
: 0702

; Routine Size: 778 bytes, Routine Base: _SMG\$CODE + 0000


```

: 445      0703 1 %SBTTL 'SMG$DRAW RECTANGLE - Draw a rectangle in a virtual display'
: 446      0704 1 GLOBAL ROUTINE SMG$DRAW_RECTANGLE (
: 447      0705 1
: 448      0706 1     DISPLAY_ID,
: 449      0707 1     TOP_LEFT_ROW,
: 450      0708 1     TOP_LEFT_COL,
: 451      0709 1     BOTTOM_RIGHT_ROW,
: 452      0710 1     BOTTOM_RIGHT_COL,
: 453      0711 1     RENDITION_SET,
: 454      0712 1     RENDITION_COMPLEMENT
: 455      0713 1 ) =
: 456      0714 1 ++
: 457      0715 1 FUNCTIONAL DESCRIPTION:
: 458      0716 1
: 459      0717 1     This routine draws a rectangle in a virtual display. Given
: 460      0718 1     two points in opposite corners of the rectangle, it determines
: 461      0719 1     the desired width and height.
: 462      0720 1
: 463      0721 1     Top left row, col *-----*
: 464      0722 1     |
: 465      0723 1     |
: 466      0724 1     |
: 467      0725 1     |
: 468      0726 1     |-----* Bottom right row, col
: 469      0727 1
: 470      0728 1     Device independent codes are placed in the display text and
: 471      0729 1     attribute buffers. Output decides which line drawing characters
: 472      0730 1     to use based on the type of terminal.
: 473      0731 1
: 474      0732 1     The internal cursor position is not changed.
: 475      0733 1
: 476      0734 1 CALLING SEQUENCE:
: 477      0735 1
: 478      0736 1     ret_status.wlc.v = SMG$DRAW_RECTANGLE (DISPLAY_ID.rl.r,
: 479      0737 1     TOP_LEFT_ROW.rl.r,
: 480      0738 1     TOP_LEFT_COL.rl.r,
: 481      0739 1     BOTTOM_RIGHT_ROW.rl.r,
: 482      0740 1     BOTTOM_RIGHT_COL.rl.r,
: 483      0741 1     [,RENDITION_SET.rl.r]
: 484      0742 1     [,RENDITION_COMPLEMENT.rl.r]
: 485      0743 1
: 486      0744 1 FORMAL PARAMETERS:
: 487      0745 1
: 488      0746 1     DISPLAY_ID.rl.r      Display id of virtual display
: 489      0747 1     TOP_LEFT_ROW.rl.r     Address of the row number of the top left
: 490      0748 1     hand corner of the rectangle.
: 491      0749 1
: 492      0750 1     TOP_LEFT_COL.rl.r    Address of the column number of the top
: 493      0751 1     left hand corner of the rectangle.
: 494      0752 1
: 495      0753 1     BOTTOM_RIGHT_ROW.rl.r    Address of the row number of the bottom right
: 496      0754 1     hand corner of the rectangle.
: 497      0755 1
: 498      0756 1     BOTTOM_RIGHT_COL.rl.r    Address of the column number of the bottom
: 499      0757 1     right hand corner of the rectangle.
: 500      0758 1
: 501      0759 1

```



```

: 502      0760 1  :   RENDITION_SET.rl.r   Optional. Each 1 bit in this parameter
: 503      0761 1  :   causes the corresponding attribute to be
: 504      0762 1  :   set in the display. (See below for list
: 505      0763 1  :   of settable attributes.)
: 506      0764 1  :
: 507      0765 1  :   RENDITION_COMPLEMENT.rl.r Optional. Each 1 bit attribute in this
: 508      0766 1  :   parameter causes the corresponding attribute
: 509      0767 1  :   to be complemented in the display. (See
: 510      0768 1  :   below for list of complementable attributes.)
: 511      0769 1  :
: 512      0770 1  :
: 513      0771 1  :   If the same bit is specified in both the RENDITION_SET parameter
: 514      0772 1  :   and in the RENDITION_COMPLEMENT parameter, the application is
: 515      0773 1  :   RENDITION_SET followed by RENDITION complement. Using these two
: 516      0774 1  :   parameters together the caller can exercise arbitrary and
: 517      0775 1  :   independent control over each attribute on a single call. On an
: 518      0776 1  :   attribute by attribute basis he can cause the following
: 519      0777 1  :   transformations:
: 520      0778 1  :
: 521      0779 1  :           SET      COMPLEMENT      Action
: 522      0780 1  :           ---      -
: 523      0781 1  :           0        0-----      Attribute unchanged.
: 524      0782 1  :           1        0              Attribute set to 'on'.
: 525      0783 1  :           0        1              Attribute set to complement of
: 526      0784 1  :           1        1              current setting.
: 527      0785 1  :           1        1              Attribute set to 'off'.
: 528      0786 1  :
: 529      0787 1  :
: 530      0788 1  :   Attributes which can be manipulated in this manner are:
: 531      0789 1  :
: 532      0790 1  :   SMG$M_BLINK displays characters blinking.
: 533      0791 1  :   SMG$M_BOLD displays characters in higher-than-normal
: 534      0792 1  :   intensity.
: 535      0793 1  :   SMG$M_REVERSE displays characters in reverse video -- that is,
: 536      0794 1  :   using the opposite default rendition of the
: 537      0795 1  :   virtual display.
: 538      0796 1  :   SMG$M_UNDERLINE displays characters underlined.
: 539      0797 1  :
: 540      0798 1  :
: 541      0799 1  :   IMPLICIT INPUTS:
: 542      0800 1  :
: 543      0801 1  :   NONE
: 544      0802 1  :
: 545      0803 1  :   IMPLICIT OUTPUTS:
: 546      0804 1  :
: 547      0805 1  :   NONE
: 548      0806 1  :
: 549      0807 1  :   COMPLETION STATUS:
: 550      0808 1  :
: 551      0809 1  :   SS$ NORMAL      Normal successful completion
: 552      0810 1  :   SMG$_INVCOL     Invalid column number
: 553      0811 1  :   SMG$_INVROW     Invalid row number
: 554      0812 1  :   SMG$_WRONUMARG  Wrong number of arguments
: 555      0813 1  :
: 556      0814 1  :   SIDE EFFECTS:
: 557      0815 1  :
: 558      0816 1  :   NONE

```



```

559 0817 1 1 1
560 0818 1 1 1
561 0819 1 1 1
562 0820 2 BEGIN
563 0821 2
564 0822 2 LOCAL
565 0823 2 BLROW, ! bottom left row
566 0824 2 BLCOL, ! bottom left column
567 0825 2 TRROW, ! top right row
568 0826 2 TRCOL, ! top right column
569 0827 2 POS, ! linear index into buffers
570 0828 2 LENGTH, ! length of line to draw
571 0829 2 TEXT_BUF : REF VECTOR [,BYTE], ! ptr to dcb text buffer
572 0830 2 ATTR_BUF : REF VECTOR [,BYTE], ! ptr to dcb attr buffer
573 0831 2 REND_CODE, ! rendition code to use
574 0832 2 DCB : REF BLOCK [,BYTE]; ! addr of display control block
575 0833 2
576 0834 2 BIND
577 0835 2 BRROW = .BOTTOM_RIGHT_ROW,
578 0836 2 BRCOL = .BOTTOM_RIGHT_COL,
579 0837 2 TLROW = .TOP_LEFT_ROW,
580 0838 2 TLCOL = .TOP_LEFT_COL;
581 0839 2
582 0840 2 LITERAL
583 0841 2 K_SET_ARG = 6; ! arg number of rend set
584 0842 2 K_COMP_ARG = 7; ! arg number of rend complement
585 0843 2
586 0844 2
587 0845 2 $SMG$GET_DCB (.DISPLAY_ID, DCB); ! get addr of virtual display
588 0846 2 ! control block
589 0847 2 TEXT_BUF = .DCB [DCB_A_TEXT_BUF];
590 0848 2 ATTR_BUF = .DCB [DCB_A_ATTR_BUF];
591 0849 2
592 0850 2 !+
593 0851 2 ! Validate arguments passed. Check for optionals.
594 0852 2 !-
595 0853 2
596 0854 2 $SMG$VALIDATE_ARGCOUNT (5, 7);
597 0855 2
598 0856 2 $SMG$VALIDATE_ROW_COL (.TLROW, .TLCOL);
599 0857 2
600 0858 2 $SMG$VALIDATE_ROW_COL (.BRROW, .BRCOL);
601 0859 2
602 0860 2 $SMG$SET_REND_CODE (K_SET_ARG, K_COMP_ARG);
603 0861 2 ! macro to use caller's args if present
604 0862 2
605 0863 2 !+
606 0864 2 ! The caller passed us two of four points needed to construct a rectangle.
607 0865 2 ! Compute the other two and make sure they are within the display.
608 0866 2 !-
609 0867 2
610 0868 2 BLROW = .BRROW;
611 0869 2 BLCOL = .TLCOL;
612 0870 2 TRROW = .TLROW;
613 0871 2 TRCOL = .BRCOL;
614 0872 2
615 0873 2 $SMG$VALIDATE_ROW_COL (.BLROW, .BLCOL);

```



```
.. 616      0874      2      $SMG$VALIDATE_ROW_COL (.TRROW, .TRCOL);
.. 617      0875
.. 618      0876      !+
.. 619      0877      ! We need to draw 4 lines, 2 horizontal and 2 vertical.
.. 620      0878      !-
.. 621      0879      ! Draw the top.
.. 622      0880      !-
.. 623      0881
.. 624      0882      BEGIN
.. 625      0883      BIND
.. 626      0884      LINE_CHAR = .DCB [DCB_A_LINE_CHAR];
.. 627      0885      MAP
.. 628      0886      LINE_CHAR : VECTOR [,BYTE];
.. 629      0887
.. 630      0888      !+
.. 631      0889      ! Reset the line characteristics vector if there was previously
.. 632      0890      ! some double high or double wide text.
.. 633      0891      !-
.. 634      0892
.. 635      0893      IF .LINE_CHAR [.TLROW] NEQ 0
.. 636      0894      THEN
.. 637      0895      $RESET_LINE (.TLROW);
.. 638      0896
.. 639      0897      !+
.. 640      0898      ! Move horizontal characters to text buffer. Horizontal chars consist
.. 641      0899      ! of a left & right segment - the left end of the line needs only a right
.. 642      0900      ! segment, the right end of the line needs a left segment, and all the
.. 643      0901      ! positions in between need both. Corners will automatically form the
.. 644      0902      ! correct characters by ORing the up/down and right/left elements.
.. 645      0903      !-
.. 646      0904
.. 647      0905      POS = $SMG$LINEAR (.TLROW, .TLCOL);
.. 648      0906      LENGTH = .TRCOL - .TLCOL + 1;
.. 649      0907
.. 650      0908      $INSERT_LINE_CHAR (BORD_M_RIGHT);
.. 651      0909      POS = .POS + 1;
.. 652      0910
.. 653      0911      INCR NUM_CHAR FROM 2 TO (.LENGTH - 1) DO
.. 654      0912      BEGIN
.. 655      0913      $INSERT_LINE_CHAR (BORD_M_HORIZ);
.. 656      0914      POS = .POS + 1;
.. 657      0915      END;
.. 658      0916
.. 659      0917      $INSERT_LINE_CHAR (BORD_M_LEFT);
.. 660      0918
.. 661      0919      !+
.. 662      0920      ! Draw the bottom.
.. 663      0921      !-
.. 664      0922
.. 665      0923      IF .LINE_CHAR [.BLROW] NEQ 0
.. 666      0924      THEN
.. 667      0925      $RESET_LINE (.BLROW);          ! reset if prev. dbl hi/wide
.. 668      0926
.. 669      0927      POS = $SMG$LINEAR (.BLROW, .BLCOL);
.. 670      0928      LENGTH = .BRCOL - .BLCOL + 1;
.. 671      0929
.. 672      0930      $INSERT_LINE_CHAR (BORD_M_RIGHT);    ! left end of line
```



```

: 673      0931      3      POS = .POS + 1;
: 674      0932      3
: 675      0933      3      INCR NUM_CHAR FROM 2 TO (.LENGTH - 1) DO ! middle of line
: 676      0934      4      BEGIN
: 677      0935      4          $INSERT_LINE_CHAR (BORD_M_HORIZ);
: 678      0936      4          POS = .POS + 1;
: 679      0937      3      END;
: 680      0938      3
: 681      0939      3      $INSERT_LINE_CHAR (BORD_M_LEFT); ! right end of line
: 682      0940      3
: 683      0941      3      !+
: 684      0942      3      !- Draw the right.
: 685      0943      3
: 686      0944      3
: 687      0945      3      !+
: 688      0946      3      !- A vertical line requires resetting multi elements in
: 689      0947      3      !- the line characteristics vector.
: 690      0948      3
: 691      0949      3
: 692      0950      3      INCR ROW FROM .TRROW TO .BRROW DO
: 693      0951      4      BEGIN
: 694      0952      4          IF .LINE_CHAR [.ROW] NEQ 0
: 695      0953      4          THEN
: 696      0954      4              $RESET_LINE (.ROW);
: 697      0955      3      END;
: 698      0956      3
: 699      0957      3      POS = $SMG$LINEAR (.TRROW, .TRCOL);
: 700      0958      3      LENGTH = .BRROW - .TRROW + 1;
: 701      0959      3
: 702      0960      3      $INSERT_LINE_CHAR (BORD_M_DOWN);
: 703      0961      3      POS = .POS + .DCB [DCB_W_NO_COLS];
: 704      0962      3
: 705      0963      3      INCR ROW FROM 2 TO (.LENGTH - 1) DO
: 706      0964      4      BEGIN
: 707      0965      4          $INSERT_LINE_CHAR (BORD_M_VERT);
: 708      0966      4          POS = .POS + .DCB [DCB_W_NO_COLS];
: 709      0967      3      END;
: 710      0968      3
: 711      0969      3      $INSERT_LINE_CHAR (BORD_M_UP);
: 712      0970      3
: 713      0971      3      !+
: 714      0972      3      !- Draw the left.
: 715      0973      3
: 716      0974      3
: 717      0975      3      INCR ROW FROM .TLROW TO .BLROW DO
: 718      0976      4      BEGIN
: 719      0977      4          IF .LINE_CHAR [.ROW] NEQ 0
: 720      0978      4          THEN
: 721      0979      4              $RESET_LINE (.ROW);
: 722      0980      3      END;
: 723      0981      3
: 724      0982      3      POS = $SMG$LINEAR (.TLROW, .TLCOL);
: 725      0983      3      LENGTH = .BLROW - .TLROW + 1;
: 726      0984      3
: 727      0985      3      $INSERT_LINE_CHAR (BORD_M_DOWN);
: 728      0986      3      POS = .POS + .DCB [DCB_W_NO_COLS];
: 729      0987      3

```



```

: 730      0988 3      INCR ROW FROM 2 TO (.LENGTH - 1) DO
: 731      0989 4      BEGIN
: 732      0990 4      $INSERT_LINE_CHAR (BORD_M_VERT);
: 733      0991 4      POS = .POS + .DCB [DCB_Q_NO_COLS];
: 734      0992 4      END;
: 735      0993 4      $INSERT_LINE_CHAR (BORD_M_UP);
: 736      0994 4      END;
: 737      0995 4      END;
: 738      0996 4      END;
: 739      0997 4      END;
: 740      0998 4      END;
: 741      0999 4      !+ See if this change should be reflected on the screen immediately.
: 742      1000 4      !-
: 743      1001 4      RETURN (SMG$$CHECK_FOR_OUTPUT_DCB (.DCB,
: 744      1002 4      0, ! no function code for this
: 745      1003 4      .TLROW));
: 746      1004 4
: 747      1005 4
: 748      1006 4
: 749      1007 4
: 750      1008 1      END;

```

! End of routine SMG\$DRAW_RECTANGLE

				OFFC 00000	.ENTRY	SMG\$DRAW_RECTANGLE, Save R2,R3,R4,R5,R6,R7,-;				
			5E	C0	AE	9E	00002	MOVAB	R8,R9,R10,R11	0704
			50	04	BC	D0	00006	MOVL	-64(SP), SP	
	04		BC	38	A0	D1	0000A	CMPL	@DISPLAY_ID, R0	0845
					06	12	0000F	BNEQ	1\$	
		11		44	A0	91	00011	CMPB	68(R0), #17	
					08	13	00015	BEQL	2\$	
		50	00000000G		8F	D0	00017	MOVL	#SMG\$_INVDIS_ID, R0	
					04	00	001E	RET		
					BC	D0	0001F	MOVL	@DISPLAY_ID, DCB	
50	0C	AE		04	10	C1	00024	ADDL3	#16, DCB, R0	0847
	3C	AE			60	D0	00029	MOVL	(R0), 60(SP)	
50	0C	AE			14	C1	0002D	ADDL3	#20, DCB, R0	0848
	38	AE			60	D0	00032	MOVL	(R0), 56(SP)	
	04	AE		38	AE	7D	00036	MOVQ	56(SP), ATTR BUF	
50		6C			05	83	0003B	SUBB3	#5, (AP), DIFF	0854
		02			50	91	0003F	CMPB	DIFF, #2	
					08	1B	00042	BLEQU	3\$	
		50	00000000G		8F	D0	00044	MOVL	#SMG\$_WRONUMARG, R0	
					04	00	004B	RET		
	24	AE		08	BC	D0	0004C	MOVL	@TOP_LEFT_ROW, 36(SP)	0856
					0D	15	00051	BLEQ	4\$	
					02	C1	00053	ADDL3	#2, DCB, R0	
24	AE	50			00	ED	00058	CMPZV	#0, #16, (R0), 36(SP)	
		60			08	18	0005E	BGEQ	5\$	
					8F	D0	00060	MOVL	#SMG\$_INVROW, R0	
					04	00	0067	RET		
					59	0C	BC	D0	@TOP_LEFT_COL, R9	
					0C	15	0006C	BLEQ	6\$	
50	0C	AE			06	C1	0006E	ADDL3	#6, DCB, R0	

59	60	10	00	ED	00073	CMPZV	#0, #16, (R0), R9	
			08	18	00078	BGEQ	7\$	
		50	00000000G	8F	D0 0007A	6\$: MOVL	#SMG\$_INVCOL, R0	
				04	00081	RET		
		28	AE	10	BC	D0 00082	7\$: MOVL	@BOTTOM_RIGHT_ROW, 40(SP)
				0D	15 00087	BLEQ	8\$	0858
28	AE	50	0C	AE	02	C1 00089	ADDL3	#2, DCB, R0
		60		10	00	ED 0008E	CMPZV	#0, #16, (R0), 40(SP)
					08	18 00094	BGEQ	9\$
		50	00000000G	8F	D0 00096	8\$: MOVL	#SMG\$_INVROW, R0	
					04	0009D	RET	
		2C	AE	14	BC	D0 0009E	9\$: MOVL	@BOTTOM_RIGHT_COL, 44(SP)
				0D	15 000A3	BLEQ	10\$	
2C	AE	50	0C	AE	06	C1 000A5	ADDL3	#6, DCB, R0
		60		10	00	ED 000AA	CMPZV	#0, #16, (R0), 44(SP)
					08	18 000B0	BGEQ	11\$
		50	00000000G	8F	D0 000B2	10\$: MOVL	#SMG\$_INVCOL, R0	
					04	000B9	RET	
		50	0C	AE	2E	C1 000BA	11\$: ADDL3	#46, DCB, R0
			34	AE	60	9A 000BF	MOVZBL	(R0), 52(SP)
			18	AE	06	D0 000C3	MOVL	52(SP), REND_CODE
					6C	91 000C8	CMPB	(AP), #6
					0A	1F 000CB	BLSSU	12\$
					AC	D5 000CD	TSTL	24(AP)
					05	13 000D0	BEQL	12\$
		18	AE	18	BC	C8 000D2	12\$: BISL2	@RENDITION_SET, REND_CODE
			07		6C	91 000D7	CMPB	(AP), #7
					0A	1F 000DA	BLSSU	13\$
					AC	D5 000DC	TSTL	28(AP)
					05	13 000DF	BEQL	13\$
		18	AE	1C	BC	CC 000E1	13\$: XORL2	@RENDITION_COMPLEMENT, REND_CODE
			58		AE	D0 000E6	MOVL	40(SP), BLROW
		20	AE	28	D0	000EA	MOVL	R9, BLCOL
			5B		AE	D0 000EE	MOVL	36(SP), TRROW
			5A		AE	D0 000F2	MOVL	44(SP), TRCOL
					58	D5 000F6	TSTL	BLROW
					0C	15 000F8	BLEQ	14\$
		50	0C	AE	02	C1 000FA	ADDL3	#2, DCB, R0
		60		10	00	ED 000FF	CMPZV	#0, #16, (R0), BLROW
					08	18 00104	BGEQ	15\$
		50	00000000G	8F	D0 00106	14\$: MOVL	#SMG\$_INVROW, R0	
					04	0010D	RET	
					AE	D5 0010E	15\$: TSTL	BLCOL
					0D	15 00111	BLEQ	16\$
		50	0C	AE	06	C1 00113	ADDL3	#6, DCB, R0
		60		10	00	ED 00118	CMPZV	#0, #16, (R0), BLCOL
					08	18 0011E	BGEQ	17\$
		50	00000000G	8F	D0 00120	16\$: MOVL	#SMG\$_INVCOL, R0	
					04	00127	RET	
					5B	D5 00128	17\$: TSTL	TRROW
					0C	15 0012A	BLEQ	18\$
		50	0C	AE	02	C1 0012C	ADDL3	#2, DCB, R0
		60		10	00	ED 00131	CMPZV	#0, #16, (R0), TRROW
					08	18 00136	BGEQ	19\$
		50	00000000G	8F	D0 00138	18\$: MOVL	#SMG\$_INVROW, R0	
					04	0013F	RET	
					5A	D5 00140	19\$: TSTL	TRCOL

5A	50	OC	AE	0C	15	00142	BLEQ	20\$			
	60		10	06	C1	00144	ADDL3	#6, DCB, R0			
				00	ED	00149	CMPZV	#0, #16, (R0), TRCOL			
				08	18	0014E	BGEQ	21\$			
			50	00000000G	8F	D0	00150	MOVL	#SMG\$_INVCOL, R0		
					04	00157	RET				
	50	OC	AE	0000004C	8F	C1	00158	ADDL3	#76, DCB, R0	0893	
	50		60	24	AE	C1	00161	ADDL3	36(SP), (R0), R0		
					60	95	00166	TSTB	(R0)		
					75	13	00168	BEQL	23\$		
	50	24	AE		01	C3	0016A	SUBL3	#1, 36(SP), R0	0895	
	52	OC	AE		06	C1	0016F	ADDL3	#6, DCB, R2		
			51		62	3C	00174	MOVZWL	(R2), R1		
			50		51	C4	00177	MULL2	R1, R0		
	52	OC	AE		04	C1	0017A	ADDL3	#4, DCB, R2		
			51		62	3C	0017F	MOVZWL	(R2), R1		
		30	AE	FF	A140	9E	00182	MOVAB	-1(R1)[R0], START_INDEX		
			50	3C	AE	D0	00188	MOVL	60(SP), TEXT_BUF		
			56	38	AE	D0	0018C	MOVL	56(SP), ATTR_BUF		
	51	OC	AE		18	C1	00190	ADDL3	#24, DCB, R1		
			57		61	D0	00195	MOVL	(R1), CHAR_BUF		
1C	BE	1C	AE	OC	06	C1	00198	ADDL3	#6, DCB, 28(SP)		
			20		00	2C	0019E	MOVC5	#0, (SP), #32, @28(SP), @START_INDEX-		
					30	BE40	001A4		[TEXT_BUF]		
					06	C1	001A7	ADDL3	#6, DCB, 28(SP)		
1C	BE	1C	AE	OC	00	2C	001AD	MOVC5	#0, (SP), 52(SP), @28(SP), @START_INDEX-		
			34		00		001B4		[ATTR_BUF]		
					30	BE46	001B7	TSTL	CHAR_BUF		
					57	D5	001B7	BEQL	22\$		
					14	13	001B9	ADDL3	#6, DCB, R6		
			56	OC	06	C1	001BB	ADDL3	#48, DCB, 28(SP)		
	1C	OC	AE		30	C1	001C0	ADDL3	#0, (SP), @28(SP), (R6), @START_INDEX-		
66	1C		6E		00	2C	001C6	MOVC5	[CHAR_BUF]		
					30	BE47	001CC		#76, DCB, R0		
					8F	C1	001CF	ADDL3	36(SP), (R0), R0		
	50	OC	AE	0000004C	24	AE	001D8	CLR8	(R0)		
	50		60		60	94	001DD	SUBL3	#1, 36(SP), R0	0905	
					01	C3	001DF	ADDL3	#6, DCB, R1		
	50	24	AE		06	C1	001E4	MOVZWL	(R1), 20(SP)		
	51	OC	AE		61	3C	001E9	MULL2	20(SP), R0		
		14	AE		61	C4	001ED	MOVAB	-1(R9)[R0], 48(SP)		
			50		14	AE	001F1	MOVL	48(SP), POS		
		30	AE	FF	A940	9E	001F7	SUBL3	R9, TRCOL, R9	0906	
			56		59	C3	001FB	INCL	LENGTH		
	59		5A		59	D6	001FF	MOVL	ATTR_BUF, R1	0908	
					04	AE	00201	BBC	#6, (POS)[R1], 24\$		
			51	6641	06	E1	00205	MOVL	TEXT_BUF, R0		
			50		08	AE	0020A	BISB2	#1, (POS)[R0]		
			6640		01	88	0020E	MOVL	ATTR_BUF, R2		
			52		04	AE	00212	MOVZBL	(POS)[R2], R0		
			50		6642	9A	00216	BICL2	#15, R0		
			50		0F	CA	0021A	MOVL	ATTR_BUF, R3		
			53		04	AE	0021D	BISB3	REND_CODE, R0, (POS)[R3]		
6643			50		18	AE	89	00221	BRB	25\$	
					13	11	00227	MOVL	TEXT_BUF, R0		
			50		08	AE	00229	MOVB	#1, (POS)[R0]		
			6640		01	90	0022D	MOVL	ATTR_BUF, R0		
			50		04	AE	00231				

6640	18	AE	40	8F	89	00235	BISB3	#64, REND_CODE, (POS)[R0]	:	0909
				56	D6	0023C	INCL	POS	:	0911
		51		01	D0	0023E	MOVL	#1, NUM_CHAR	:	0913
				3D	11	00241	BRB	29\$:	
		52	04	AE	D0	00243	MOVL	ATTR_BUF, R2	:	
1F	6642			06	E1	00247	BBC	#6, (POS)[R2], 27\$:	
	50		08	AE	D0	0024C	MOVL	TEXT_BUF, R0	:	
	6640			05	88	00250	BISB2	#5, (POS)[R0]	:	
	53		04	AE	D0	00254	MOVL	ATTR_BUF, R3	:	
	50			6643	9A	00258	MOVZBL	(POS)[R3], R0	:	
	50			0F	CA	0025C	BICL2	#15, R0	:	
	54		04	AE	D0	0025F	MOVL	ATTR_BUF, R4	:	
6644	50		18	AE	89	00263	BISB3	REND_CODE, R0, (POS)[R4]	:	
				13	11	00269	BRB	28\$:	
	50		08	AE	D0	0026B	MOVL	TEXT_BUF, R0	:	
	6640			05	90	0026F	MOVB	#5, (POS)[R0]	:	
	50		04	AE	D0	00273	MOVL	ATTR_BUF, R0	:	
6640	18	AE	40	8F	89	00277	BISB3	#64, REND_CODE, (POS)[R0]	:	0914
				56	D6	0027E	INCL	POS	:	0911
BF	51			59	F2	00280	AOBLSS	LENGTH, NUM_CHAR, 26\$:	0917
	51		04	AE	D0	00284	MOVL	ATTR_BUF, R1	:	
1F	6641			06	E1	00288	BBC	#6, (POS)[R1], 30\$:	
	50		08	AE	D0	0028D	MOVL	TEXT_BUF, R0	:	
	6640			04	88	00291	BISB2	#4, (POS)[R0]	:	
	52		04	AE	D0	00295	MOVL	ATTR_BUF, R2	:	
	50			6642	9A	00299	MOVZBL	(POS)[R2], R0	:	
	50			0F	CA	0029D	BICL2	#15, R0	:	
	53		04	AE	D0	002A0	MOVL	ATTR_BUF, R3	:	
6643	50		18	AE	89	002A4	BISB3	REND_CODE, R0, (POS)[R3]	:	
				13	11	002AA	BRB	31\$:	
	50		08	AE	D0	002AC	MOVL	TEXT_BUF, R0	:	
	6640			04	90	002B0	MOVB	#4, (POS)[R0]	:	
	50		04	AE	D0	002B4	MOVL	ATTR_BUF, R0	:	
6640	18	AE	40	8F	89	002B8	BISB3	#64, REND_CODE, (POS)[R0]	:	0923
50	0C	AE	0000004C	8F	C1	002BF	ADDL3	#76, DCB, R0	:	
			00	B048	95	002C8	TSTB	@0(R0)[BLROW]	:	
				5F	13	002CC	BEQL	33\$:	0925
	50		FF	A8	9E	002CE	MOVAB	-1(R8), R0	:	
	50		14	AE	C4	002D2	MULL2	20(SP), R0	:	
52	0C	AE		04	C1	002D6	ADDL3	#4, DCB, R2	:	
		51		62	3C	002DB	MOVZWL	(R2), R1	:	
	1C	AE		FF	A140	9E	MOVAB	-1(R1)[R0], START_INDEX	:	
		50		3C	AE	D0	MOVL	60(SP), TEXT_BUF	:	
		57		38	AE	D0	MOVL	56(SP), ATTR_BUF	:	
51	0C	AE		18	C1	002EC	ADDL3	#24, DCB, R1	:	
	10	AE		61	D0	002F1	MOVL	(R1), CHAR_BUF	:	
14	AE	20		00	2C	002F5	MOVCS	#0, (SP), #32, 20(SP), @START_INDEX-	:	
				1C	BE40	002FB		[TEXT_BUF]	:	
14	AE	34	AE		00	2C	MOVCS	#0, (SP), 52(SP), 20(SP), @START_INDEX-	:	
				1C	BE47	00305		[ATTR_BUF]	:	
				10	AE	D5	TSTL	CHAR_BUF	:	
				13	13	0030B	BEQL	32\$:	
		57	10	AE	D0	0030D	MOVL	CHAR_BUF, R7	:	
14	AE	00	6E	0C	AE	30	ADDL3	#48, DCB, (SP)	:	
		BE	6E		00	2C	MOVCS	#0, (SP), @0(SP), 20(SP), @START_INDEX[R7]	:	
				1C	BE47	0031D			:	
	50	0C	AE	0000004C	8F	C1	ADDL3	#76, DCB, R0	:	
							32\$:		:	

		00	B048	94	00329	CLRB	20(R0)[BLROW]			
	50	FF	A8	9E	0032D	33\$:	MOVAB	-1(R8), R0	0927	
	50	14	AE	C4	00331		MULL2	20(SP), R0		
51	20	AE	01	C3	00335		SUBL3	#1, BLCOL, R1		
56		50	51	C1	0033A		ADDL3	R1, R0, POS		
50	2C	AE	20	AE	C3	0033E	SUBL3	BLCOL, 44(SP), R0	0928	
		59	01	A0	9E	00344	MOVAB	1(R0), LENGTH		
	51	04	AE	D0	00348		MOVL	ATTR_BUF, R1	0930	
1F	6641		06	E1	0034C		BBC	#6, (POS)[R1], 34\$		
	50	08	AE	D0	00351		MOVL	TEXT_BUF, R0		
	6640		01	88	00355		BISB2	#1, (POS)[R0]		
	52	04	AE	D0	00359		MOVL	ATTR_BUF, R2		
	50		6642	9A	0035D		MOVZBL	(POST[R2]), R0		
	50		0F	CA	00361		BICL2	#15, R0		
	53	04	AE	D0	00364		MOVL	ATTR_BUF, R3		
6643	50	18	AE	89	00368		BISB3	REND_CODE, R0, (POS)[R3]		
		13	11	0036E			BRB	35\$		
	50	08	AE	D0	00370	34\$:	MOVL	TEXT_BUF, R0		
	6640		01	90	00374		MOVB	#1, (POS)[R0]		
	50	04	AE	D0	00378		MOVL	ATTR_BUF, R0		
6640	18	AE	40	8F	89	0037C	BISB3	#64, -REND_CODE, (POS)[R0]		
		56	D6	00383		35\$:	INCL	POS	0931	
	51		01	D0	00385		MOVL	#1, NUM_CHAR	0933	
		3D	11	00388			BRB	39\$		
	52	04	AE	D0	0038A	36\$:	MOVL	ATTR_BUF, R2	0935	
1F	6642		06	E1	0038E		BBC	#6, (POS)[R2], 37\$		
	50	08	AE	D0	00393		MOVL	TEXT_BUF, R0		
	6640		05	88	00397		BISB2	#5, (POS)[R0]		
	53	04	AE	D0	0039B		MOVL	ATTR_BUF, R3		
	50		6643	9A	0039F		MOVZBL	(POST[R3]), R0		
	50		0F	CA	003A3		BICL2	#15, R0		
	54	04	AE	D0	003A6		MOVL	ATTR_BUF, R4		
6644	50	18	AE	89	003AA		BISB3	REND_CODE, R0, (POS)[R4]		
		13	11	003B0			BRB	38\$		
	50	08	AE	D0	003B2	37\$:	MOVL	TEXT_BUF, R0		
	6640		05	90	003B6		MOVB	#5, (POS)[R0]		
	50	04	AE	D0	003BA		MOVL	ATTR_BUF, R0		
6640	18	AE	40	8F	89	003BE	BISB3	#64, -REND_CODE, (POS)[R0]		
		56	D6	003C5		38\$:	INCL	POS	0936	
BF	51		59	F2	003C7	39\$:	AOSLSS	LENGTH, NUM_CHAR, 36\$	0933	
	51	04	AE	D0	003CB		MOVL	ATTR_BUF, R1	0939	
1F	6641		06	E1	003CF		BBC	#6, (POS)[R1], 40\$		
	50	08	AE	D0	003D4		MOVL	TEXT_BUF, R0		
	6640		04	88	003D8		BISB2	#4, (POS)[R0]		
	52	04	AE	D0	003DC		MOVL	ATTR_BUF, R2		
	50		6642	9A	003E0		MOVZBL	(POST[R2]), R0		
	50		0F	CA	003E4		BICL2	#15, R0		
	53	04	AE	D0	003E7		MOVL	ATTR_BUF, R3		
6643	50	18	AE	89	003EB		BISB3	REND_CODE, R0, (POS)[R3]		
		13	11	003F1			BRB	41\$		
	50	08	AE	D0	003F3	40\$:	MOVL	TEXT_BUF, R0		
	6640		04	90	003F7		MOVB	#4, (POS)[R0]		
	50	04	AE	D0	003FB		MOVL	ATTR_BUF, R0		
6640	18	AE	40	8F	89	003FF	BISB3	#64, -REND_CODE, (POS)[R0]		
	57	FF	AB	9E	00406	41\$:	MOVAB	-1(R11), ROW	0950	
			78	11	0040A		BRB	44\$		
50	0C	AE	0000004C	8F	C1	0040C	42\$:	ADDL3	#76, DCB, R0	0952

				00	B047	95	00415		TSTB	@0(R0)[ROW]	
					69	13	00419		BEQL	44\$	
		50		FF	A7	9E	0041B		MOVAB	-1(R7), R0	
		50		14	AE	C4	0041F		MULL2	20(SP), R0	0954
	52	0C			04	C1	00423		ADDL3	#4, DCB, R2	
					62	3C	00428		MOVZWL	(R2), R1	
		2C		FF	A140	9E	0042B		MOVAB	-1(R1)[R0], START_INDEX	
				3C	AE	D0	00431		MOVL	60(SP), TEXT_BUF	
	51	1C		38	AE	D0	00435		MOVL	56(SP), ATTR_BUF	
		0C			18	C1	0043A		ADDL3	#24, DCB, R1	
	20	20			61	D0	0043F		MOVL	(R1), CHAR_BUF	
14	AE				00	2C	00443		MOVCS	#0, (SP), #32, 20(SP), @START_INDEX-	
				2C	BE40		00449			[TEXT_BUF]	
				1C	AE	C1	0044C		ADDL3	ATTR_BUF, START_INDEX, 16(SP)	
14	AE	10	AE	2C	AE				MOVCS	#0, 7SP, 52(SP), 20(SP), @16(SP)	
	34	AE			00	2C	00453				
					10	BE	0045A				
					20	AE	D5	0045C	TSTL	CHAR_BUF	
					16	13	0045F		BEQL	43\$	
					20	AE	C1	00461	ADDL3	CHAR_BUF, START_INDEX, 28(SP)	
	1C	AE	2C	AE	30	C1	00468		ADDL3	#48, DCB, 16(SP)	
14	AE	10	OC	AE	00	2C	0046E		MOVCS	#0, (SP), @16(SP), 20(SP), @28(SP)	
	10	BE			1C	BE	00475				
					04C	8F	C1	00477	ADDL3	#76, DCB, R0	
	50	OC	AE	000000	00	B047	94	00480	CLRB	@0(R0)[ROW]	
	83				28	AE	F3	00484	AOBLEQ	40(SP), ROW, 42\$	0950
					FF	AB	9E	00489	MOVAB	-1(R11), R7	0957
					14	AE	C4	0048D	MULL2	20(SP), R7	
					FF	AA47	9E	00491	MOVAB	-1(TRCOL)[R7], POS	
	5A	28			5B	C3	00496		SUBL3	TRROW, 40(SP), R10	0958
					01	AA	9E	0049B	MOVAB	1(R10), LENGTH	
					04	AE	D0	0049F	MOVL	ATTR_BUF, R1	0960
	1F		6641		06	E1	004A3		BBC	#6, 7POS[R1], 45\$	
			50		08	AE	D0	004A8	MOVL	TEXT_BUF, R0	
			6640		08	88	004AC		BISB2	#8, 7POS[R0]	
			52		04	AE	D0	004B0	MOVL	ATTR_BUF, R2	
			50		6642	9A	004B4		MOVZBL	(POS)[R2], R0	
			50		0F	CA	004B8		BICL2	#15, R0	
			53		04	AE	D0	004BB	MOVL	ATTR_BUF, R3	
6643			50		18	AE	89	004BF	BISB3	REND_CODE, R0, (POS)[R3]	
					13	11	004C5		BRB	46\$	
			50		08	AE	D0	004C7	MOVL	TEXT_BUF, R0	45\$:
			6640		08	90	004CB		MOVAB	#8, 7POS[R0]	
			50		04	AE	D0	004CF	MOVL	ATTR_BUF, R0	
6640		18	AE		40	8F	89	004D3	BISB3	#64, REND_CODE, (POS)[R0]	
			56		14	AE	C0	004DA	ADDL2	20(SP), POS	0961
			51		01	D0	004DE		MOVL	#1, ROW	0963
					3F	11	004E1		BRB	50\$	
			52		04	AE	D0	004E3	MOVL	ATTR_BUF, R2	47\$:
	1F		6642		06	E1	004E7		BBC	#6, 7POS[R2], 48\$	
			50		08	AE	D0	004EC	MOVL	TEXT_BUF, R0	
			6640		0A	88	004F0		BISB2	#10, (POS)[R0]	
			53		04	AE	D0	004F4	MOVL	ATTR_BUF, R3	
			50		6643	9A	004F8		MOVZBL	(POS)[R3], R0	
			50		0F	CA	004FC		BICL2	#15, R0	
			54		04	AE	D0	004FF	MOVL	ATTR_BUF, R4	
6644			50		18	AE	89	00503	BISB3	REND_CODE, R0, (POS)[R4]	
					13	11	00509		BRB	49\$	

		50	08	AE	D0	0050B	48\$:	MOVL	TEXT_BUF, R0		
		6640		0A	90	0050F		MOVB	#10, -(POS)[R0]		
		50	04	AE	D0	00513		MOVL	ATTR_BUF, R0		
6640	18	AE	40	8F	89	00517		BISB3	#64, -REND CODE, (POS)[R0]		
		56	14	AE	C0	0051E	49\$:	ADDL2	20(SP), POS	0966	
BD		51		59	F2	00522	50\$:	AOBLS	LENGTH, ROW, 47\$	0963	
		51	04	AE	D0	00526		MOVL	ATTR_BUF, R1	0969	
1F		6641		06	E1	0052A		BBC	#6, (POS)[R1], 51\$		
		50	08	AE	D0	0052F		MOVL	TEXT_BUF, R0		
		6640		02	88	00533		BISB2	#2, (POS)[R0]		
		52	04	AE	D0	00537		MOVL	ATTR_BUF, R2		
		50		6642	9A	0053B		MOVZBL	(POST[R2], R0		
		50		0F	CA	0053F		BICL2	#15, R0		
		53	04	AE	D0	00542		MOVL	ATTR_BUF, R3		
6643		50	18	AE	89	00546		BISB3	REND_CODE, R0, (POS)[R3]		
				13	11	0054C		BRB	52\$		
		50	08	AE	D0	0054E	51\$:	MOVL	TEXT_BUF, R0		
		6640		02	90	00552		MOVB	#2, (POS)[R0]		
		50	04	AE	D0	00556		MOVL	ATTR_BUF, R0		
6640	18	AE	40	8F	89	0055A		BISB3	#64, -REND CODE, (POS)[R0]		
5A	24	AE		01	C3	00561	52\$:	SUBL3	#1, 36(SP), ROW	0975	
				67	11	00566		BRB	55\$		
50	0C	AE	0000004C	8F	C1	00568	53\$:	ADDL3	#76, DCB, R0	0977	
			00	B04A	95	00571		TSTB	@0(R0)[ROW]		
				58	13	00575		BEQL	55\$		
		50	FF	AA	9E	00577		MOVAB	-1(R10), R0	0979	
		50	14	AE	C4	0057B		MULL2	20(SP), R0		
52	0C	AE		04	C1	0057F		ADDL3	#4, DCB, R2		
		51		62	3C	00584		MOVZWL	(R2), R1		
	2C	AE	FF	A140	9E	00587		MOVAB	-1(R1)[R0], START_INDEX		
		50	3C	AE	D0	0058D		MOVL	60(SP), TEXT_BUF		
		57	38	AE	D0	00591		MOVL	56(SP), ATTR_BUF		
51	0C	AE		18	C1	00595		ADDL3	#24, DCB, R1		
		5B		61	D0	0059A		MOVL	(R1), CHAR_BUF		
14	AE	20		00	2C	0059D		MOVCS	#0, (SP), #32, 20(SP), @START_INDEX-		
				2C	BE40	005A3			[TEXT_BUF]		
14	AE	34	AE		00	2C	005A6	MOVCS	#0, (SP), 52(SP), 20(SP), @START_INDEX-		
				2C	BE47	005AD			[ATTR_BUF]		
				5B	D5	005B0		TSTL	CHAR_BUF		
				0E	13	005B2		BEQL	54\$		
		57	0C	AE	30	C1	005B4	ADDL3	#48, DCB, R7		
14	AE	67		00	2C	005B9		MOVCS	#0, (SP), (R7), 20(SP), @START_INDEX-		
				2C	BE4B	005BF			[CHAR_BUF]		
50	0C	AE	0000004C	8F	C1	005C2	54\$:	ADDL3	#76, DCB, R0		
			00	B04A	94	005CB		CLRB	@0(R0)[ROW]		
95		5A		58	F3	005CF	55\$:	AOBLEQ	BLROW, ROW, 53\$	0975	
		56	30	AE	D0	005D3		MOVL	48(SP), POS	0982	
		58	24	AE	C2	005D7		SUBL2	36(SP), R8	0983	
		59	01	A8	9E	005DB		MOVAB	1(R8), LENGTH		
		51	04	AE	D0	005DF		MOVL	ATTR_BUF, R1	0985	
1F		6641		06	E1	005E3		BBC	#6, (POS)[R1], 56\$		
		50	08	AE	D0	005E8		MOVL	TEXT_BUF, R0		
		6640		08	88	005EC		BISB2	#8, (POS)[R0]		
		52	04	AE	D0	005F0		MOVL	ATTR_BUF, R2		
		50		6642	9A	005F4		MOVZBL	(POST[R2], R0		
		50		0F	CA	005F8		BICL2	#15, R0		
		53	04	AE	D0	005FB		MOVL	ATTR_BUF, R3		

6643	50	18	AE	89	005FF	BISB3	REND_CODE, R0, (POS)[R3]	:	
	50	13	11	00605	BRB	57\$:		
	6640	08	AE	D0	00607	56\$:	MOVL	TEXT_BUF, R0	
	50	08	90	0060B	MOVB	#8, (POS)[R0]	:		
6640	18	04	AE	D0	0060F	MOVL	ATTR_BUF, R0	:	
	56	40	8F	89	00613	BISB3	#64, REND_CODE, (POS)[R0]	:	
	51	14	AE	C0	0061A	57\$:	ADDL2	20(SP), POS	
			01	D0	0061E	MOVL	#1, ROW	:	
			3F	11	00621	BRB	61\$:	
	52	04	AE	D0	00623	58\$:	MOVL	ATTR_BUF, R2	
1F	6642		06	E1	00627	BBC	#6, (POS)[R2], 59\$:	
	50	08	AE	D0	0062C	MOVL	TEXT_BUF, R0	:	
	6640		0A	88	00630	BISB2	#10, (POS)[R0]	:	
	53	04	AE	D0	00634	MOVL	ATTR_BUF, R3	:	
	50	6643	9A	00638	MOVZBL	(POST[R3], R0	:		
	50		0F	CA	0063C	BICL2	#15, R0	:	
	54	04	AE	D0	0063F	MOVL	ATTR_BUF, R4	:	
6644	50	18	AE	89	00643	BISB3	REND_CODE, R0, (POS)[R4]	:	
			13	11	00649	BRB	60\$:	
	50	08	AE	D0	0064B	59\$:	MOVL	TEXT_BUF, R0	
	6640		0A	90	0064F	MOVB	#10, (POS)[R0]	:	
	50	04	AE	D0	00653	MOVL	ATTR_BUF, R0	:	
6640	18	40	8F	89	00657	BISB3	#64, REND_CODE, (POS)[R0]	:	
	56	14	AE	C0	0065E	60\$:	ADDL2	20(SP), POS	
BD	51		59	F2	00662	61\$:	AOBLSS	LENGTH, ROW, 58\$	
	51	04	AE	D0	00666	MOVL	ATTR_BUF, R1	:	
1F	6641		06	E1	0066A	BBC	#6, (POS)[R1], 62\$:	
	50	08	AE	D0	0066F	MOVL	TEXT_BUF, R0	:	
	6640		02	88	00673	BISB2	#2, (POS)[R0]	:	
	52	04	AE	D0	00677	MOVL	ATTR_BUF, R2	:	
	50	6642	9A	0067B	MOVZBL	(POST[R2], R0	:		
	50		0F	CA	0067F	BICL2	#15, R0	:	
	53	04	AE	D0	00682	MOVL	ATTR_BUF, R3	:	
6643	50	18	AE	89	00686	BISB3	REND_CODE, R0, (POS)[R3]	:	
			13	11	0068C	BRB	63\$:	
	50	08	AE	D0	0068E	62\$:	MOVL	TEXT_BUF, R0	
	6640		02	90	00692	MOVB	#2, (POS)[R0]	:	
	50	04	AE	D0	00696	MOVL	ATTR_BUF, R0	:	
6640	18	40	8F	89	0069A	BISB3	#64, REND_CODE, (POS)[R0]	:	
		24	AE	DD	006A1	63\$:	PUSHL	36(SP)	
			7E	D4	006A4	CLRL	-(SP)	:	
		14	AE	DD	006A6	PUSHL	DCB	:	
00000000G	00		03	FB	006A9	CALLS	#3, SMG\$\$CHECK_FOR_OUTPUT_DCB	:	
			04	006B0	RET		:	1008	

0986

0988

0990

0991

0988

0994

1005

1003

; Routine Size: 1713 bytes, Routine Base: _SMG\$CODE + 030A

; 751 1009 1 !<BLF/PAGE>

0356 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY